1cd

قدم به قدم

میکرو کنترلر برای مبتدیان



آرش کوزه کنانی

قدم به قدم

میکروکنترلر برای مبتدیان

آرش کوزه کنانی

اطلاعات الکترونیکی من قبل از نوشتن این کتاب در حد دانستن تفاوت بین مقاومت و خازن بود. تا اینکه تصمیم گرفتم یک وسیله بازی درست کنم. ابتدا برای کنترل کردن قطع و وصل شدن چهار کلید، از سیم‌هایی که از زیر به کلیدهای کیبورد یک کامپیوتر وصل می‌کردم استفاده می‌کردم. بدین ترتیب که به طور مثال کلید شماره ۱ را به سیم‌های زیر دکمه **A** وصل کردم و در برنامه کامپیوتر تعریف کردم هر وقت کلید **A** فشار داده شد (یعنی زمانی که کلید ۱ فشار داده می‌شد) فلان کار را انجام بده و همین‌طور برای کلیدهای دیگر...

در حین انجام این کار بودم که شنیدم با استفاده از وسیله‌ای به نام میکروکنترلر می‌توانم راحت‌تر وسیله بازی را درست کنم.

کتابی که در دست دارید مراحلی است که خودم طی کردم تا کار کردن با یک میکروکنترلر را یاد بگیرم.

با تشکر از:

روشن الهیاری

ابوذر هداوند

مریم کوزه کنانی

حمید صدقی‌نژاد

فریبا محمدپور

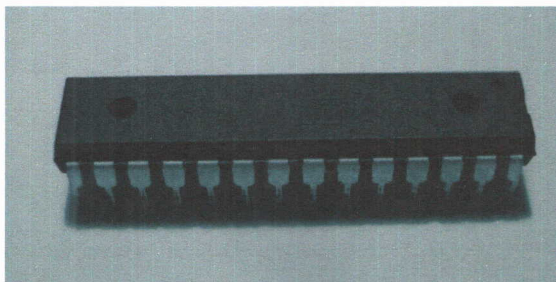
مژده کوزه کنانی

ویرایش سوم

مقدمه

میکروکنترلر قطعه‌ای الکترونیکی است که می‌توان آن را برنامه‌ریزی کرد. بدین ترتیب که برنامه مورد نظر در ابتدا توسط یک کامپیوتر شخصی در محیط یک نرم‌افزار مخصوص نوشته می‌شود. سپس توسط کابلی که از کامپیوتر به میکروکنترلر وصل شده است، برنامه به میکروکنترلر منتقل می‌شود. پس از آن میکرو قادر است مستقل از کامپیوتر برنامه را اجرا کند.

در این راهنما ما از میکروکنترلر Mega8 از سری میکروکنترلرهای AVR که ساده و ارزان قیمت می‌باشد استفاده می‌کنیم.

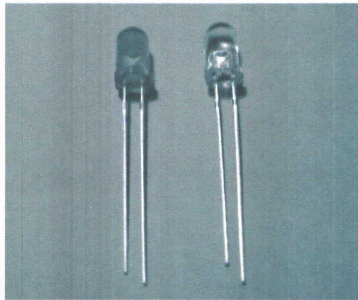


تصویر شماره ۱

در این قسمت چند نمونه از کارهایی را که می‌خواهیم با میکروکنترلر انجام دهیم را شرح می‌دهم:

- به یکی از پایه‌های میکروکنترلر یک لامپ LED وصل می‌کنیم. میکروکنترلر را طوری برنامه‌نویسی می‌کنیم تا لامپ با یک سرعت ثابت چشمک بزند.

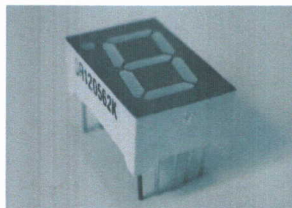
لامپ LED یک دیود است که با برقرار شدن جریانی در حدود ۳ ولت از خود نور ساطع می‌کند. دارای دو پایه می‌باشد. پایه بلندتر مثبت و پایه کوتاهتر منفی می‌باشد.



تصویر شماره ۲

- دو کلید فشاری به میکرو متصل می‌کنیم و میکرو را طوری برنامه‌ریزی می‌کنیم تا با فشار دادن یکی از کلیدها سرعت چشمک‌زن LED زیاد و با فشار دادن کلید دیگر سرعت چشمک‌زن کم شود.
- با وصل کردن یک 7-segment به میکرو به طور محدود سرعت چشمک‌زن LED را نمایش می‌دهیم.

دم در یک آسانسور ایستاده‌اید و عجله دارید تا زودتر آسانسور به طبقه شما بیاید در این لحظات معمولاً به جایی که طبقه آسانسور را نشان می‌دهد نگاه می‌کنید شما در حال نگاه کردن به یک 7-segment می‌باشید. در پشت یک چراغ قرمز شمارنده‌ای که ثانیه‌های باقی مانده به سبز شدن چراغ را نشان می‌دهد یک 7-segment می‌باشد.

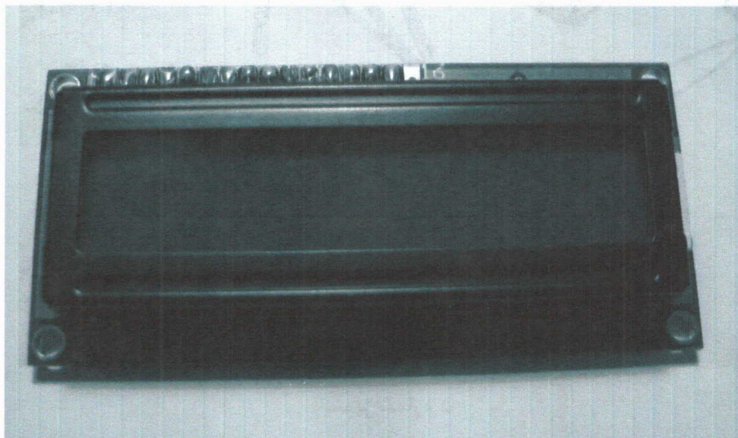


تصویر شماره ۳

- و در انتها توسط یک LCD که به میکروکنترلر وصل می‌کنیم قادر به نمایش اطلاعات

نوشتاری در سیستم خواهیم بود.

وقتی با یک ماشین حساب کار می‌کنید اعداد بر روی یک LCD نمایش داده می‌شوند. صفحه کامپیوتری ساعتتان نیز یک LCD می‌باشد.



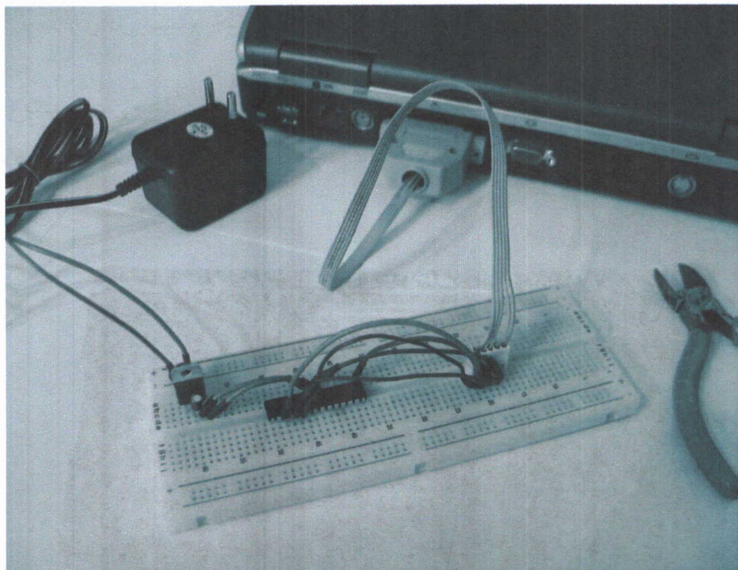
تصویر شماره ۴

برای راه‌اندازی یک میکروکنترلر به سه وسیله اصلی احتیاج داریم:

۱- کابلی که میکرو را به کامپیوتر وصل کند.

۲- منبع تغذیه‌ای که میکرو با آن کار کند.

۳- بوردی که بتوانیم میکروکنترلر و منبع تغذیه را بر روی آن سوار کنیم.



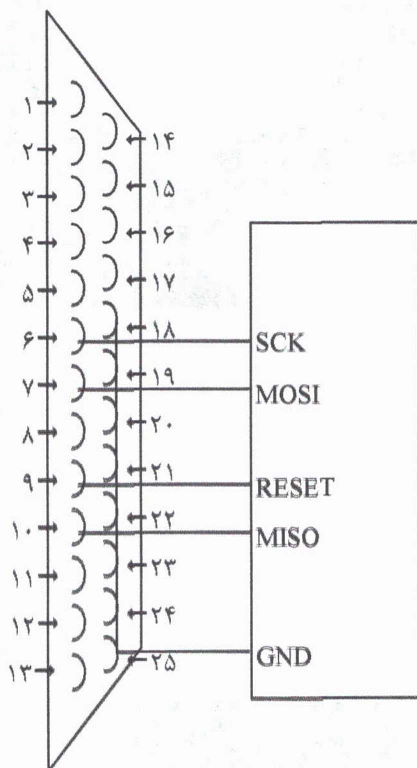
تصویر شماره ۵

کابل رابط بین کامپیوتر و میکروکنترلر (Programmer)

این کابل‌ها بر اساس آنکه به کدام خروجی کامپیوتر (پورت USB و یا پورت Printer) وصل می‌شوند و یا در درونشان از IC Buffer استفاده می‌شود یا خیر، انواع مختلفی دارند در این قسمت دستور ساخت ساده‌ترین نوع کابل Programmer را شرح می‌دهیم.

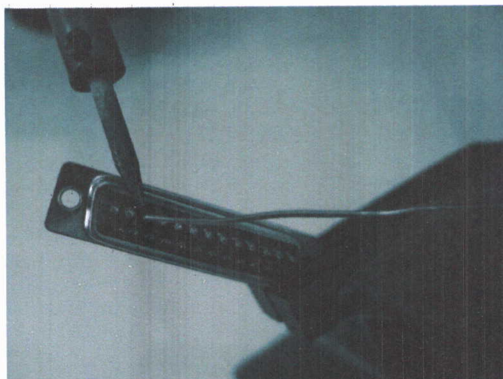
برای ساخت چنین رابطی قطعات زیر مورد نیاز است:

- یک سیم فلت ۵ رشته‌ای به طول تقریبی ۴۰ سانتی‌متر (می‌توانید از ۵ رشته اول یک سیم ۱۰ رشته نیز استفاده کنید)
- یک سوکیت ۲۵ پین نرگی به همراه کاور
- کانکتور مخابراتی ۵ پین (نرگی و مادگی)



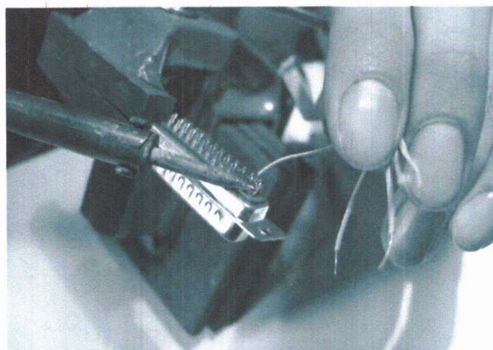
تصویر شماره ۶

سوکیت ۲۵ پین (از سمتی که قرار است سیم‌ها به آن لحیم شود) را روبه‌روی خودتان قرار دهید. (برای راحتی کار می‌توانید از یک گیره رومیزی کوچک استفاده کنید) سپس درون گودی پین‌هایی که مطابق شکل باید سیم به آن لحیم شود را مقداری لحیم داغ می‌کنیم. توجه کنید چون پین‌های ۱۸تا ۲۵ همگی به سیم GND وصل هستند همهٔ آنها را توسط یک سیم لخت به یکدیگر لحیم می‌کنیم و یکی از آنها را به سیم پنجم فلت و یا همان سیم GND وصل می‌کنیم.



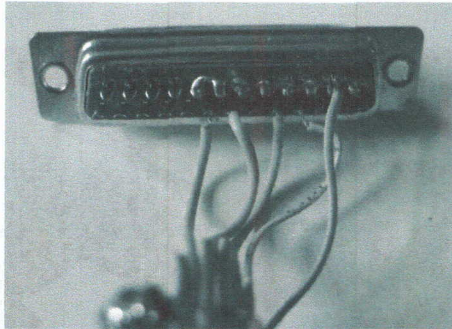
تصویر شماره ۷

برای وصل کردن سیم فلت به سوکیت ۴۰ cm از سیم را جدا می‌کنیم و در هر دو طـرف سیم به عمق ۳ تا ۴ سانتی‌متر سیم‌ها را از هم جدا می‌کنیم و نوک هر کدام از سـیم‌هـا را بـه اندازهٔ ۳ mm لخت می‌کنیم. سپس سر سیم‌ها را به ترتیبی که در شـکل نشـان داده شـده بـه پین‌هایی که قبلاً در آن لحیم ریخته‌ایم قـرار مـی‌دهـیم و لحـیم را دوبـاره توسـط هویـه آب می‌کنیم و نوک سیم را در درون آن فشار می‌دهیم.



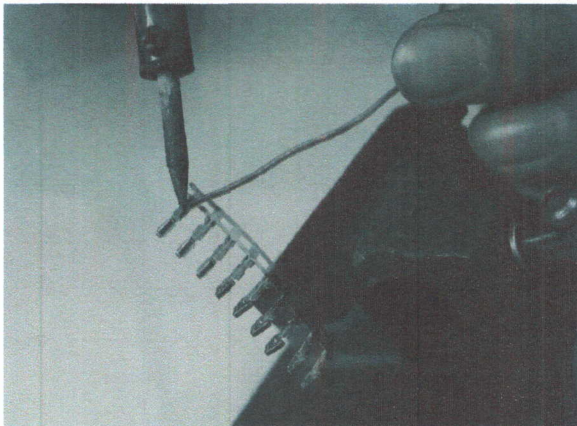
تصویر شماره ۸

در این قسمت توجه کنید سیم‌ها رشته رشته نشوند و با پین‌های پهلویی اتصال نداشته باشند.



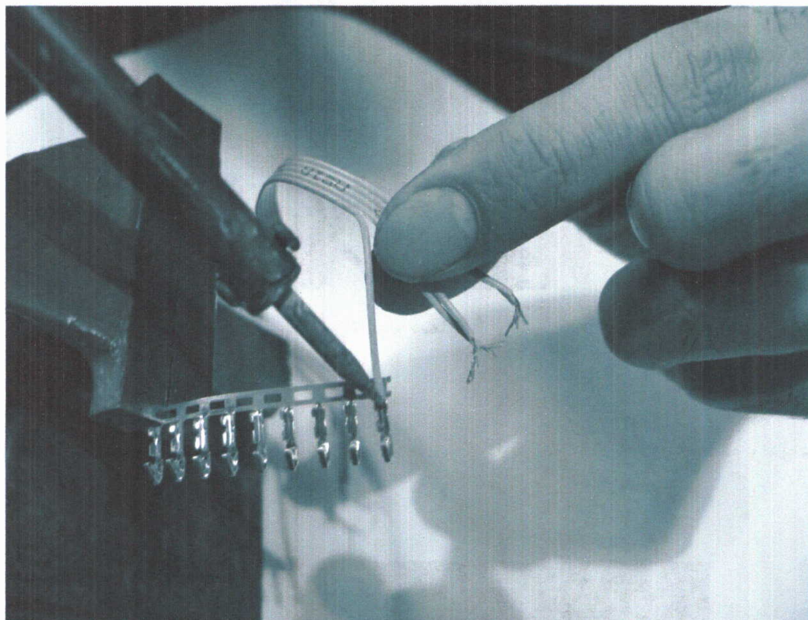
تصویر شماره ۹

اما طرف دیگر سیم فلت را به ترتیب زیر به مادگی یک کانکتور مخابراتی ۵ پین وصل می‌کنیم.
پین‌های فلزی را همان طور که بر روی ریل قرار دارند در جایی محکم کرده و در داخل ۵ تا از آنها مقداری لحیم می‌ریزیم.



تصویر شماره ۱۰

سپس سرلخت سیم‌های فلت را داخل شیارهای پین‌های فلزی قرار داده و به وسیله هویه لحیم را دوباره آب می‌کنیم تا سیم درون پین بچسبد.



تصویر شماره ۱۱

زائده‌های بغل پین را روی سیم فشار می‌دهیم تا سیم درون پین محکم شود سپس با یک دم باریک مقداری با پین بازی می‌کنیم تا از بیخ از ریل جدا شود.

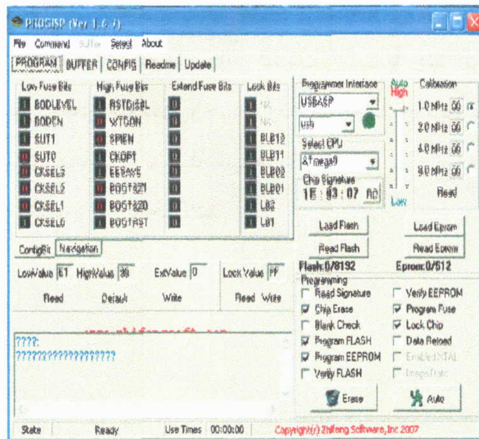
پین‌ها را به ترتیبی که در تصویر شماره ۶ نامگذاری کرده‌ایم داخل مادگی کانکتور مخابراتی قرار می‌دهیم. (توجه کنید که پشت پین‌ها باید در سمت سوراخ‌های کانکتور قرار گیرد.



تصویر شماره ۱۲

کابل Programmer ما آماده است.

- توضیحی که داده شد در مورد programmer با پورت printer(parallel) است,ولی امکان دارد بعضی از کامپیوتر ها این پورت را نداشته باشد و شما باید از programmer با پورت usb استفاده کنید.شما قادر به ساخت این نوع programmer نیستید و باید از مراکز خریدی که آدرس آ ن ذکر شده است تهیه کنید.
- همراه این programmer, CD وجود دارد که باید آن را روی کامپیوتر نصب کنید.بعد از نصب آن کامپیوتر را restart کنید و پورت usb programmer را به کامپیوتر متصل کنید.فایل PROGISP را از برنامه نصب شده بار کنید.صفحه ای با این شکل می باشد



در این صفحه باید option های مختلف را بر حسب نیازتنظیم کنید,که الان به چند مورد که برای شروع کار لازم است اشاره میکنیم.

۱.درprogrammer ,interface ←انتخاب می کنید USBASP.usb

۲.در select cpu ← نوع میکرو را انتخاب می کنید.

۳.در programming ← علامت می زنید chiperase,programflash,programeeprom,programfuse,lockchip

برای program کردن میکرو , اول دکمه Loadflash را زده, فایل Hex برنامه ای که می خواهید باز کنید

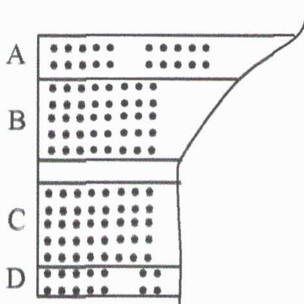
وسپس با دکمه AUTO برنامه را به میکرو منتقل کنید.

حال باید بورد مناسبی برای سوار کردن میکروکنترلر و منبع تغذیه انتخاب کنیم. بوردها انواع مختلف دارند.

۱) بورد نقطه‌ای: که قطعات درون سوراخ‌های آن لحیم شده و توسط سیم به هم وصل می‌شوند.

۲) بوردهایی که بر روی فیبر مخصوص توسط خودمان طراحی می‌شوند.

۳) Bread Board یا برد تخته‌ای که ما در این قسمت کتاب با چنین بردی کار می‌کنیم. در چنین بردی سوراخ‌های متعددی وجود دارد که قطعات را باید داخل سوراخ‌ها فرو کرده و سپس ارتباط قطعات را بوسیلهٔ سیم‌های که داخل سوراخ‌ها می‌کنیم برقرارر می‌شود. به تصویر شماره ۱۳ توجه کنید.



تصویر شماره ۱۳

- تمام نقاطی که در امتداد دو خط قرمز و آبی به صورت افقی در محدودهٔ A و D قرار دارند از زیر به هم وصل می‌باشند (توجه کنید نقاط بالا وپایین به یکدیگر وصل نیستند.
- هر ۵ نقطه‌ای که به صورت عمودی در محدودهٔ B و C قرار دارند از زیر به هم وصل می‌باشند (توجه کنید نقاط پهلویی به هم وصل نیستند و نقاط در محدودهٔ B با نقاط در محدودهٔ C با هم اتصال ندارند.

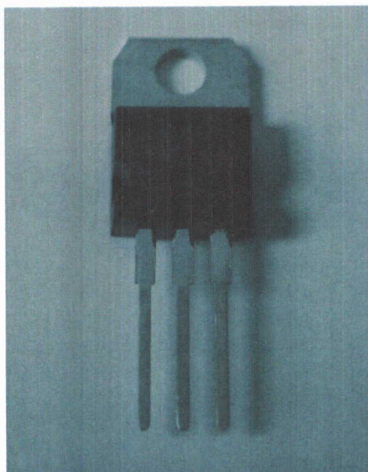
برای سیم‌هایی که بر روی بورد بکار می‌برید یا از سیم مخصوص Bread bord و یا از سیم مفتولی استفاده کنید تا به راحتی درون بورد فرو برود.

منبع تغذیه

یک باطری یک منبع تغذیه می‌باشد، یک آداپتور یک منبع تغذیه می‌باشد و تفاوت منابع تغذیه با یکدیگر در ولتاژ و شدت جریان می‌باشد. اما برای راه‌اندازی یک میکروکنترلر Mega 8 به ولتاژی بین ۴/۵ تا ۵/۵ ولت به صورت جریان DC (مستقیم) احتیاج داریم و در اینجا یک روش ساده برای بوجود آوردن چنین ولتاژی را شرح می‌دهیم.

برای ساخت یک منبع تغذیه ۵ ولت به قطعات زیر احتیاج داریم:

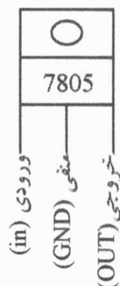
- یک آداپتور یا خروجی DC که ولتاژ خروجی آن در حدود ۸ ولت می‌باشد. (بدین منظور از کلمه حدود استفاده کردم که مثلاً اگر یک آداپتور ۱۵ ولت هم در خانه پیدا کردید بدرد کار ما می‌خورد. محدودهٔ مورد استفاده بین ۶ تا ۲۰ ولت می‌باشد.
- یک رگلاتور ۷۸۰۵: رگلاتور قطعه‌ای است که هر ورودی با ولتاژ بالای ۵ ولت را به خروجی با ولتاژ ۵ ولت تبدیل می‌کند.



تصویر شماره ۱۴

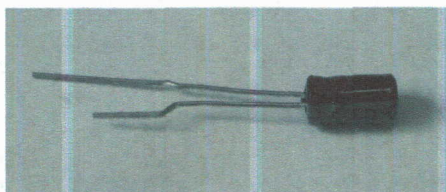
توجه کنید اگر بجای رگلاتور ۷۸۰۵ از یک رگلاتور ۷۸۰۶ استفاده کنید خروجی شما ۶ ولت خواهد بود و همین طور برای رگلاتور ۷۸۱۵ خروجی شما ۱۵ ولت می‌باشد. برای بدست آوردن یک ولتاژ ۳ ولتی از یک رگلاتور LF33 استفاده کنید.

ترتیب قرار گرفتن پایه‌های رگلاتور ۷۸۰۵ به صورت زیر می‌باشد:



تصویر شماره ۱۵

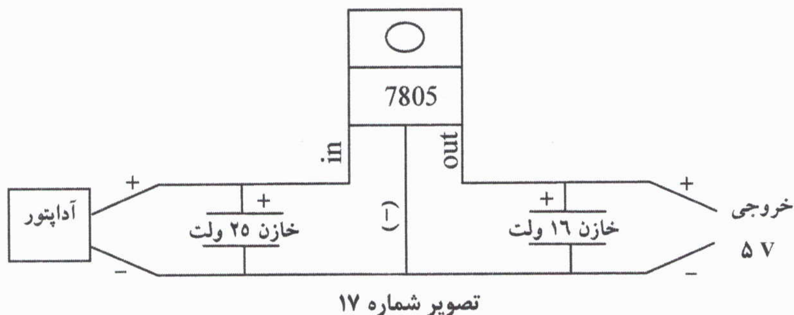
- یک خازن ۲۵ ولت، ۱۰۰ میکروفاراد
- یک خازن ۱۶ ولت، ۴۷ میکروفاراد



تصویر شماره ۱۶

توجه کنید که این خازن‌ها از نوع الکترولیت بوده و دارای پایه مثبت و منفی می‌باشند. اگر به تصویر شماره ۱۶ دقت کنید خواهید دید که پایه مثبت از پایه منفی خازن بلندتر بود و در امتداد پایه منفی خازن علامت (-) بر روی بدنه خازن نقش بسته است.

اما آداپتور، رگلاتور و خازن‌ها را باید به ترتیب نقشه زیر به هم وصل کنید:



تصویر شماره ۱۷

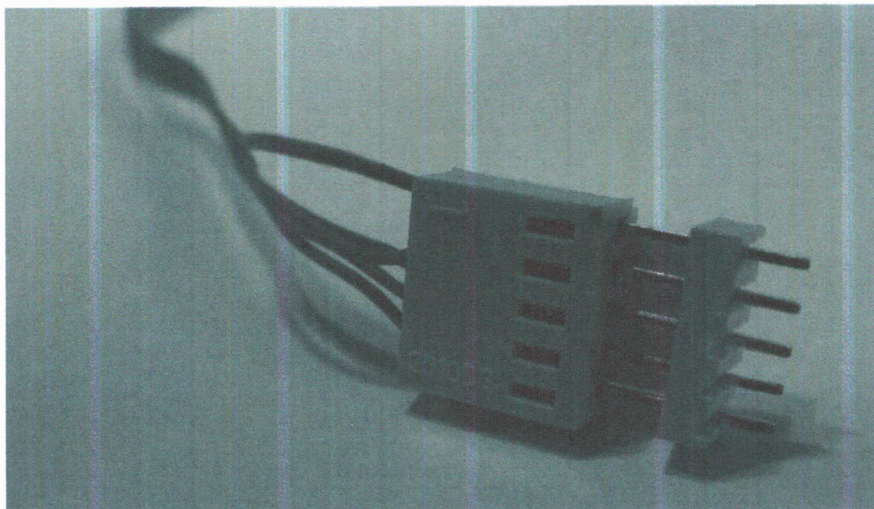
و تصویر شماره ۱۸ نحوه نصب قطعات بر روی برد مطابق با نقشه را نشان می‌دهد:

تصویر شماره ۱۸

برای آنکه بفهمید کدام سیم آداپتور مثبت و کدام سیم منفی می‌باشد multimetr را بر روی ۲۰ V تنظیم کنید و در حالی که آداپتور به برق وصل است دو شاخه multimetr را به دو سیم خروجی آداپتور وصل کنید اگر ولتاژ نشان داده شده مثبت باشد نشان دهنده آن است که سیم مثبت را به شاخه قرمز multimetr وصل کرده‌اید و اگر مقدار منفی باشد شما سیم منفی را به شاخه قرمز وصل کرده‌اید. حال داری یک منبع تغذیه ۵ ولت بر روی بورد هستیم.

برای امتحان کردن آن کافیست دو سیمی را که از خازن ۱۶ ولت خارج شده را به دو شاخه multimetr وصل کرده و ولتاژ ۵ V را مشاهده کنیم.

به سراغ کابل رابط رفته و نرگی مخابراتی ۵ پین را درون مادگی کانکتور فرو می‌کنیم:

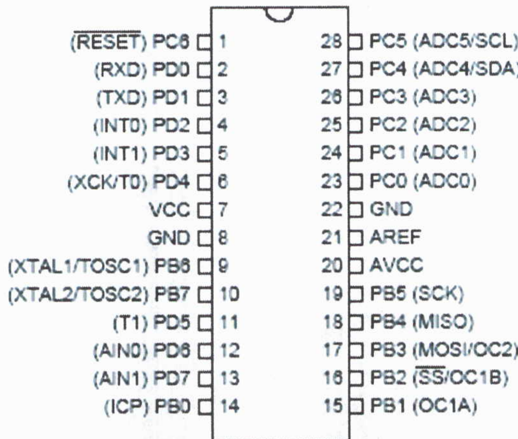


تصویر شماره ۱۹

سپس به سراغ بورد رفته و میکروکنترلر را طوری بر روی آن نصب می‌کنیم که پایه‌های

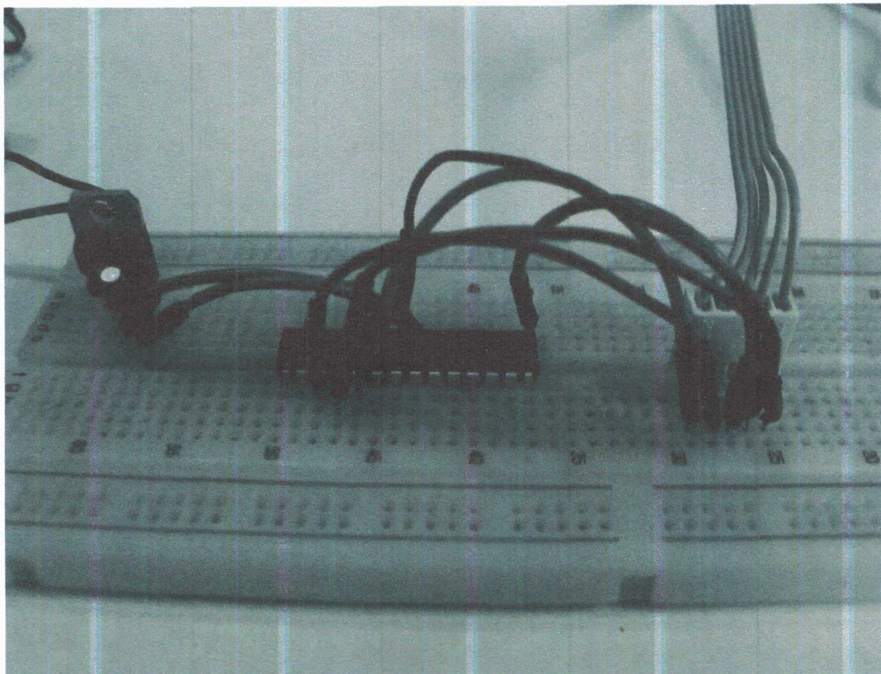
آن در دو طرف شیار وسط بورد قرار گیرد و سر میکروکنترلر در سمت راست ما باشد (سر میکروکنترلر را توسط یک نیم دایره کوچک تو رفته می‌توانید تشخیص دهید) و نرگی کانکتور را به فاصلهٔ حدود ۳ سانتی‌متر از میکروکنترلر را بورد فرو می‌کنیم.

اگر یادتان باشد (تصویر شماره ۶) بر روی هر کدام از ۵ سیم فلت یک نام نهاده بودیم به ترتیب SKC، MOSI، RESET و MISO و GND اگر به نقشه میکروکنترلر دقت کنید:



تصویر شماره ۲۰

خواهید دید ۵ پایه بر روی میکروکنترلر به همین نام‌ها وجود دارند. حال باید هر کدام از پایه‌های نرگی کانکتور را بوسیلهٔ سیم به پایهٔ مورد نظر از میکروکنترلر وصل کنیم.



تصویر شماره ۲۱

بدین ترتیب که سیم اول را داخل سوراخ روبروی پایه SKC از میکروکنترلر فرو می‌کنیم و انتهای دیگر آن را روبروی کانکتور ۵ پین مقابل سیمی که SKC نام نهاده بودیم فرو می‌کنیم و همین‌طور برای بقیه سیم‌ها از طرف دیگر برای آنکه جریان را از منبع تغذیه به میکروکنترلر برسانیم از انتهای خازن ۱۶ ولت (منبع تغذیه) سیم مثبت را به پایهٔ VCC و سیم منفی را به پایه GND از میکروکنترلر وصل می‌کنیم.

توجه کنید که یک سیم هم از کانکتور مخابراتی به پایه GND میکروکنترلر وصل شده و ما در حقیقت در روبروی پایه GND دو سیم در سوراخ‌های بودر فرو کرده‌ایم.

در این مرحله دقت کنید:

۱- کانکتور ۵ پین کاملاً در بورد فرو رفته باشد.

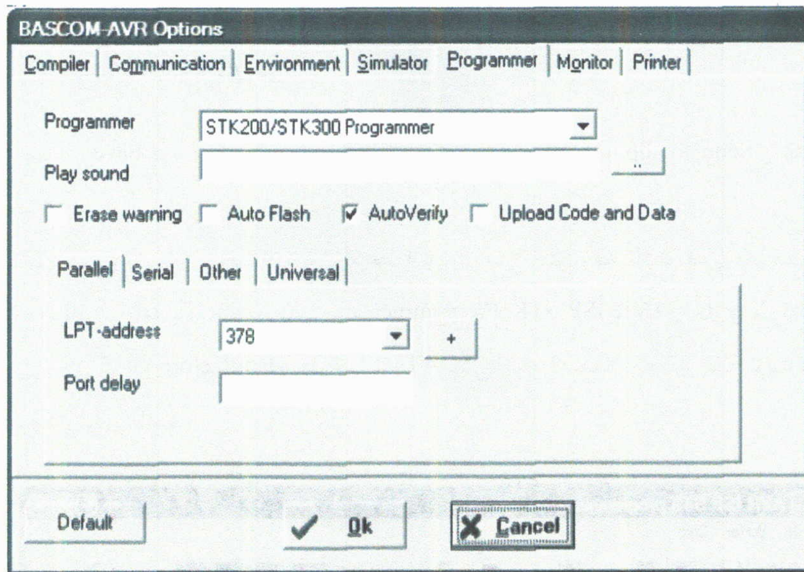
۲- سیم‌هایی که از کانکتور به میکروکنترلر وصل کرده‌اید کاملاً در بورد فرو رفته باشد ترتیبشان صحیح بوده و با یکدیگر اتصالی نداشته باشند.

بوسیله یک multimetr می‌توانید از صحیح بودن اتصالات مطمئن شوید. بدین ترتیب که multimetr را بر روی بوق اتصال کوتاه تنظیم کنید. مثلاً یک شاخه را بر روی پین(SKC) از سوکیت ۲۵ پین گذاشته و شاخه دیگر را بر روی پایه (SKC) میکروکنترلر قرار می‌دهیم. اگر همهٔ اتصالات بدرستی برقرار باشد multimetr بوق می‌زند و همین‌طور برای سیم‌های دیگر.

با وصل کردن سوکیت ۲۵ پین به پورت Printer کامپیوتر سخت‌افزار ما کامل شده حال به سراغ نرم‌افزار می‌رویم:

ابتدا نرم‌افزار Bascome را بر روی کامپیوتر install می‌کنیم و برنامه را اجرا می‌کنیم.

نرم‌افزار کابل رابطی را که ما ساخته‌ایم را بنام STK200/300 می‌شناسد. به همین منظور در منوی اصلی بر روی option رفته و بر روی گزینهٔ programmer کلیک می‌کنیم و در داخل منو نام Programmer را به STK200/300 تغییر داده و همین‌طور Port را Parallel انتخاب می‌کنیم.



تصویر شماره ۲۲

به منوی اصلی برگشته و از ستون File گزینه New را انتخاب می‌کنیم. صفحهٔ خالی که باز شده است جایی است که می‌توانیم در آن برنامه بنویسیم قبل از هر کاری باید ببینیم تا تمامی قسمت‌های سخت‌افزار بدرستی کار می‌کنند. بدین منظور باید امتحان کنیم آیا نرم‌افزار قادر به شناسایی میکروکنترلر می‌باشد یا خیر؟

خط زیر را تایپ می‌کنیم:

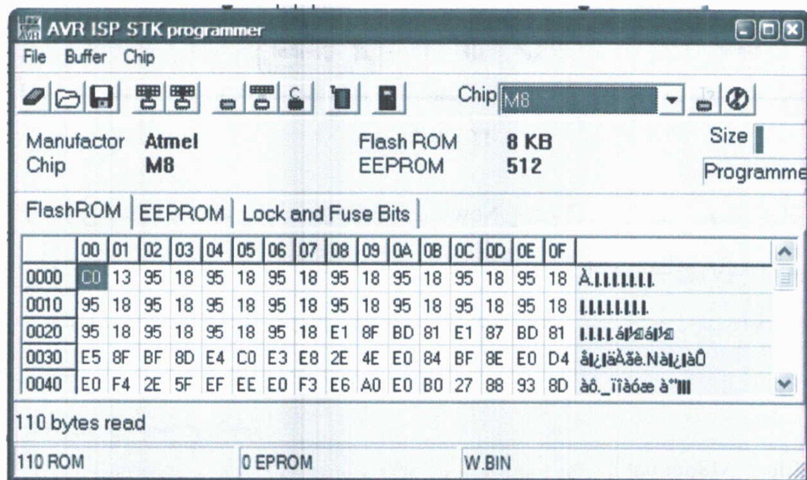
```
$ regfile= "M8def.dat"
```

M8def نامی است که بر روی میکروکنترلر Mega8 گذاشته شده است و ما با نوشتن این خط در ابتدای هر برنامه برای نرم‌افزار توضیح می‌دهیم که ما با میکروکنترلر Mega8 کار خواهیم کرد. حال به ستون Program رفته و گزینهٔ Compile را انتخاب می‌کنیم.

هر برنامه قبل از فرستاده شدن به میکروکنترلر ابتدا باید Compile شود تا اشکالات احتمالی آن برطرف شده آماده ارسال شود.

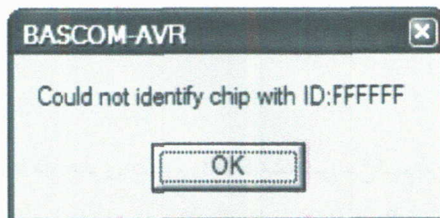
پس از Save کردن برنامه با نام دلخواه از ستون Program گزینه Send to dip را انتخاب کنید.

اگر تمامی کارهای سخت‌افزاری و نرم‌افزاری بدرستی پیش رفته باشد بدون دریافت هیچ پیغام خطایی برنامه یک منوی جدید بنام AVR ISP STK Programmer را بازی می‌کند که در روبروی گزینهٔ Manufactore کلمهٔ ATMEL نقش بسته که همان نام کارخانهٔ تولیدکننده میکروکنترلر می‌باشد.



تصویر شماره ۲۳

در غیر این صورت اگر در جایی اشکالی وجود داشته باشد و یا میکروکنترلر شما سالم نباشد ابتدا پنجره پیغام زیر



تصویر شماره ۲۴

و سپس پنجرهٔ پیغام ذیل



تصویر شماره ۲۵

نمایش داده می‌شود و در منوی AVRISP STK Program روبـروی گزینـهٔ Manufactor کلمهٔ unknown نقش می‌بندد.

باید دوباره چک کنیم آیا همهٔ اتصالات از کامپیوتر تا میکرو بدرستی وصل شـده‌انـد. آیـا منبع تغذیه بدرستی ولتاژ ۵ ولت را بـه میکـرو ارسـال مـی‌کنـد آیـا در گزینـهٔ Option نـوع Programmer بدرستی انتخاب شده است و...

برای چک کردن لازم نیست تمامی مراحل نرم‌افـزاری را دوبـاره تکـرار کنیـد در منـوی AVRISP STK Program دکمه‌ای قـرار دارد بنـام Chip Idenity کافیـسـت آن را فشـار دهید اگر اشکال برطرف شده باشد روبروی گزینهٔ Manufactor کلمهٔ ATMEL نقش می‌بندد.

در این مرحله می‌خواهیم بوسیلهٔ میکروکنترلر یک لامپ LED را روشن مـی‌کنـیم. ابتـدا باید پایه‌ای را که می‌خواهیم LED را به آن وصل کنیم خروجـی تعریـف کنـیم. بـرای مثـال

می‌خواهیم از پایهٔ PB2 استفاده کنیم. برای آنکه پایهٔ PB2 را خروجی تعریف کنیم از دستور زیر استفاده می‌کنیم:

```
DDRb.2=1
```

با گذاشتن حرف b در جلوی دستور DDR مشخص می‌کنیم می‌خواهیم راجع به پورت B اطلاعات دهیم (به مجموعه پایه‌هایی که نام B دارند Port B می‌گوییم) و با گذاشتن عدد ۲ شماره پورت را معرفی می‌کنیم.

عدد ۱ مشخص می‌کند پایه خروجی می‌باشد و عدد صفر نشان می‌دهد می‌خواهیم از پایه به عنوان ورودی استفاده کنیم:

حال توسط دستور:

```
Portb.2=1
```

جریان را در پایه PB2 برقرار می‌کنیم.

و اما برای منتقل کردن برنامه به میکروکنترلر

```
$ regfile= "M8def.dat"
DDRb.2=1
Portb.2=1
```

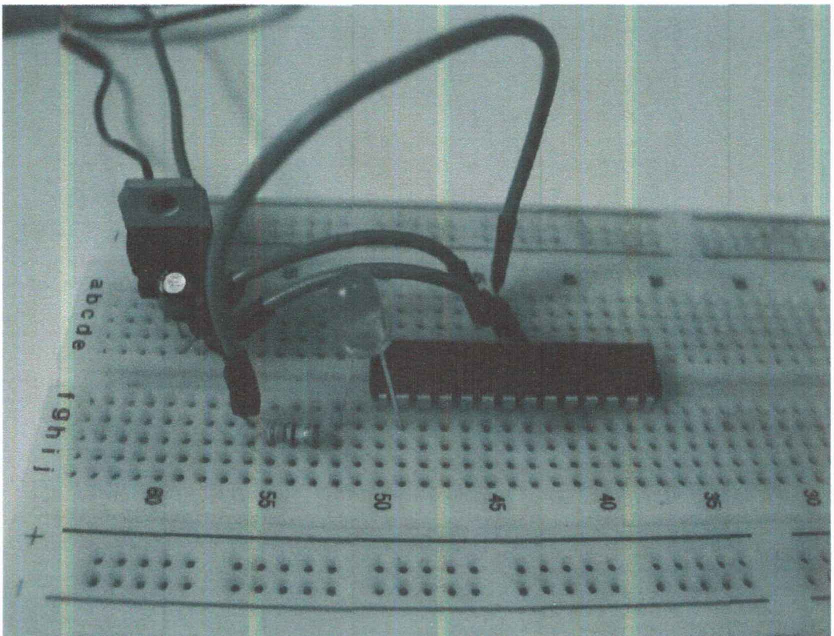
ابتدا برنامه را Compile کرده سپس گزینهٔ Send to chip را انتخاب می‌کنیم منوی AVR ISP STK Program ظاهر می‌شود بر روی دکمه Aoto Program کلیک کنید تا برنامه بر روی میکروکنترلر منتقل شود اگر همه چیز بدرستی انجام شده باشد در پایین منو گزینهٔ Verified ok نقش می‌بندد و اگر برنامه بدرستی منتقل نشده باشد پنجره پیغام ذیل باز می‌شود.



تصویر شماره ۲۶

که نشان دهندهٔ آن است که برنامه نوشته شده در نرم‌افزار با برنامه منتقل شده به میکروکنترلر مغایرت دارد. این اختلاف گاهی به خاطر تغییرات ولتاژ بوجود می‌آید که با امتحان کردن دوبارهٔ Aoto program chip حل می‌شود ولی اگر دوباره پیغام ظاهر شد باید چک کنید، سیم‌ها به هم اتصال نداشته باشند و بدرستی وصل شده‌اند، آیا منبع تغذیه به درستی جریان V ۵ را به میکرو منتقل می‌کند و... .

حال به سراغ بورد می‌رویم برای آنکه جریان خارج شده از پایه میکروکنترلر را کاهش دهیم یک مقاومت ۳۳۰ اهمی با LED سری می‌کنیم.پایه مثبت LED را به میکرو و پایه منفی را بوسیله مقاومت به GND وصل می کنیم. مجموعهٔ منبع تغذیه، میکروکنترلر، LED و مقاومت در شکل زیر نمایش داده شده است.



تصویر شماره ۲۷

پس از آنکه برنامه بدرستی به میکروکنترلر منتقل شد کافیست کانکتور ۵ پین را از بورد جدا کنید تا برنامه اجرا شود خواهید دید که LED روشن می‌شود. حال اگر بخواهیم برنامه‌ای بنویسیم که LED چشمک بزند.

```
$ regfile= "M8def.dat"
   Ddrb.2=1
   Do
   Portb.2=1
   Waitms 100
   Portb.2=0
   Waitms 100
   Loop
```

دستوراتی که بین Do و Loop قرار می‌گیرند مرتباً تکرار می‌شوند مگر آنکه برای حلقه یک شرط بگذاریم تا پس از برقرار شدن شرط برنامه از حلقه خارج شود مانند:

```
A=1
Do
A=A+1
Loop until A<10
```

این حلقه ۹ بار تکرار شده و زمانی که A=۱۰ می‌شود از حلقه خارج می‌شود.

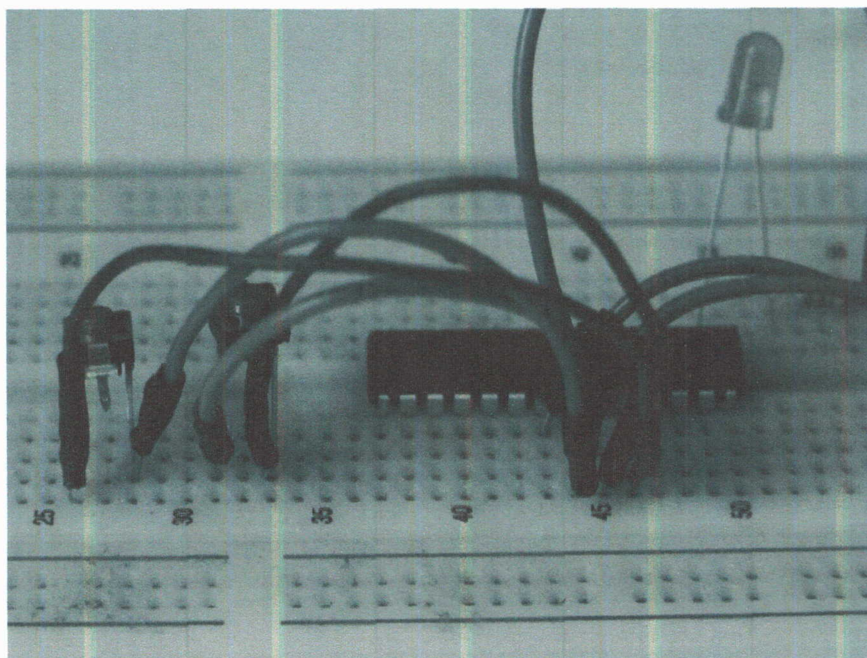
دستور Waitms به اندازهٔ مقداری که به آن می‌دهیم در اجرای برنامه تأخیر ایجاد می‌کند برای مثال: Waitms 100 به اندازهٔ ۱۰۰ میلی ثانیه تأخیر ایجاد می‌کند (که البته این مقدار خیلی دقیق نیست).

با اجرای این برنامه مشاهده می‌کنید که LED چشمک می‌زند و البته می‌توانیم با کم و زیاد کردن عدد جلوی Waitms سرعت چشمک‌زن LED را کم و زیاد می‌کنیم.

در این قسمت می‌خواهیم دو کلید به میکروکنترلر اضافه کنیم تا بوسیله این دو کلید بتوانیم در حین اجرای برنامه سرعت چشمک‌زن LED را کم و زیاد کنیم.

کلیدی که انتخاب می‌کنیم باید از نوع شاسی باشد (فشاری). یعنی تا زمانی که کلید را فشار می‌دهیم جریان برقرار شده و زمانی که دستمان را از روی کلید برمی‌داریم جریان قطع می‌شود.

ما به دلخواه پایه‌های PB6 و PB7 را برای اتصال کلیدها انتخاب می‌کنیم. یکی از پایه‌های کلید اول را به PB6 و پایه دیگر را به GND و یکی از پایه‌های کلید دوم را به PB7 و پایه دیگر را به GND وصل می‌کنیم.



تصویر شماره ۲۸

و دستورات زیر را وارد میکروکنترلر می‌کنید:

```
$ regfile= "M8def.dat"
    Ddrb.2=1
    Ddrb.6=1
    Ddrb.7=1
    Portb.6=1
    Portb.7=1
Dim S AS Integer
    S=100
Do
    Portb.2=1
    Waitms S
    Portb.2=0
    Waitms S
    IF Pinb.6=0  Then   S=S+10
    IF Pinb.7=0  Then  S=S-10
Loop
```

زمانی که از یک متغیر مانند S در برنامه‌ای استفاده می‌کنید باید برای نرم‌افزار تعریف کنید این متغیر از چه نوعی می‌باشد در خط

DIM S AS Integer

ما S را مقداری عددی، صحیح و مثبت تعریف کرده‌ایم.

هر پایهٔ میکرو دارای یک حافظه مخصوص می‌باشد که به آن Pin می‌گویند که مقداری که هر لحظه در این Pin ریخته می‌شود بیانگر وضعیت جریان در آن پایه می‌باشد برای نمونه ما پایهٔ PB6 را توسط دستور DDR b.6=1 خروجی تعریف کردیم و با نوشتن دستور Port b.6=1 ولتاژ را در پایه برقرار کردیم تا زمانی که ما کلید شاسی مربوط به پایه PB6 را فشار نداده‌ایم مدار باز بوده و جریانی از پایه PB6 خارج نمی‌شود در این حالت عددی که در حافظهٔ Pinb.6 ریخته می‌شود عدد ۱ می‌باشد یعنی (Pinb.6=1) ولی اگر کلید را فشار دهیم و مدار را ببندیم جریان از پایهٔ PB6 خارج شده و عددی که در حافظه Pinb.6 ریخته می‌شود عدد صفر می‌باشد یعنی Pinb.6=0. حالا می‌توانید حدس بزنید

وقتی از دستور:

```
IF Pinb.6=0  Then  S = S+10
```

در برنامه استفاده می‌کنید به میکروکنترلر دستور می‌دهید تا زمانی که کلید مربوط به پایه PB6 فشار داده شده است ۱۰ واحد به S اضافه کند و همین‌طور زمانی که کلید مربوط به پایه PB7 فشار داده شده ۱۰ واحد از S کم کند. به همین نسبت سرعت چشمک‌زن LED کم و زیاد می‌شود.

حال با اضافه کردن یک 7-segment به میکروکنترلر می‌توانیم افزایش و یا کاهش سرعت چشمک‌زن LED را توسط اعدادی که در 7-segment نمایش داده می‌شود را ببینیم.

یک 7-segmentاز هفت LED اصلی تشکیل شده که اگر با ترتیبی خاص آنها را روشن کنیم می‌توانیم اعداد بین صفر تا ۹ را نمایش دهیم. به دو نوع

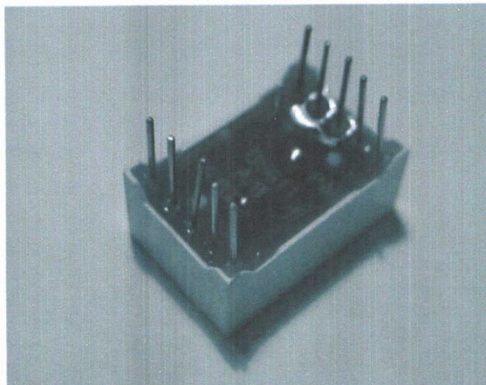
- آند مشترک
- کاتد مشترک

تقسیم می‌شوند.

آند مشترک یعنی پایه مثبت همهٔ LEDها مشترک می‌باشند که این پایه آند مشترک را باید به قطب مثبت منبع تغذیه (VCC) وصل کرد و تک تک پایه‌های کاتد را باید به پایه‌های میکروکنترلر وصل کنیم. و در کاتد مشترک پایهٔ منفی همهٔ LEDها مشترک بوده که قطب منفی (GND) منبع تغذیه وصل می‌شوند و پایه‌های آند تک تک LEDها به پایه‌های میکروکنترلر وصل می‌شود.

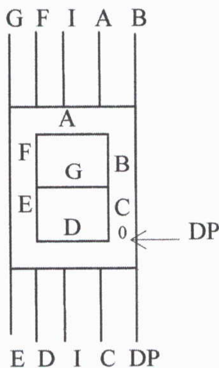
برای تشخیص آنکه یک 7-segment آند مشترک یا کاتد مشترک می‌باشد از یک منبع تغذیه بین ۳ تا ۵ ولت استفاده کنید معمولاً پایه‌های وسط دو طرف 7-segment پایه‌های مشترک می‌باشند. با وصل کردن مثلاً قطب مثبت منبع تغذیه به پایه وسط و قطب منفی منبع تغذیه به هر کدام از پایه‌های دیگر اگر LED روشن شد تشخیص می‌دهید 7-segment آند

مشترک می‌باشد و همین‌طور برعکس برای کاتد مشترک.



تصویر شماره ۲۹

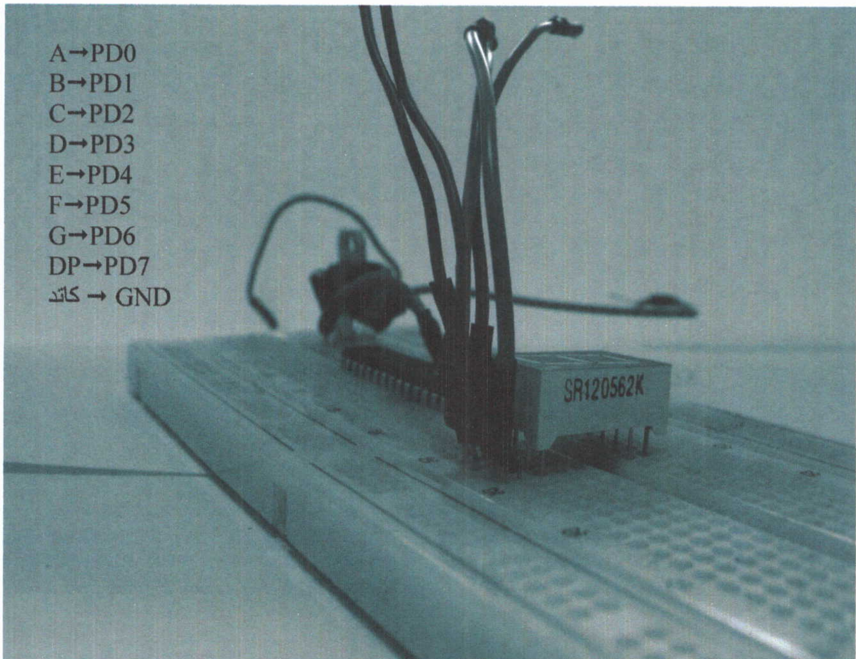
در شکل زیر ترتیب قرارگرفتن پایه‌های یک 7-segment کاتد مشترک را نشان می‌دهد:



تصویر شماره ۳۰

توجه کنید اگر بخواهید برای مثال عدد ۱ را توسط 7-segment نمایش دهید در حالی که پایه کاتد را به قطب منفی منبع تغذیه وصل کرده‌اید پایه‌های B و C را به قطب مثبت (VCC) منبع تغذیه وصل کنید و یا برای نمایش عدد ۲ پایه‌های E، D، G، B و A را به منبع تغذیه وصل کنید.

حال برای آنکه بتوانیم اعداد را توسط میکروکنترلر نمایش دهیم یک 7-segment کاتد مشترک را بر روی بورد نصب کرده و پایه‌های آن را به ترتیب زیر به پایه‌های Port D وصل می‌کنیم:



تصویر شماره ۳۱

حال برای مثال برای آنکه 7-segment عدد ۱ را نمایش دهد کد زیر را تایپ می‌کنیم:

```
$ regfile= "M8def.dat"
    Ddrd.1=1
    Ddrd.2=1
    Portd.1=1
    Portd.2=1
```

برای نمایش تمامی اعداد باید تمام پایه‌های D را خروجی تعریف کنیم. برای این کار می‌توانیم بجای آنکه تک تک DDR هر پایه را برابر با یک بنویسیم از دستور زیر استفاده کنیم:

```
Ddrd= &b11111111
```

اعداد به ترتیب از سمت راست به چپ تعیین کننده وضعیت پایه PD0، PD1، PD2، PD3، PD4، PD5، PD6 و PD7 می‌باشند شبیه اینکار را می‌توانیم زمانی که می‌خواهیم به چند پایه ولتاژ بدهیم استفاده کنیم مثلاً تصور کنید می‌خواهید در پایهٔ PB2 و PB7 جریان برقرار کنید باید بنویسید:

```
Portb.2=1
Portb.7=1
```

اما می‌توانید بجای نوشتن این دو خط از خط زیر استفاده کنید:

```
Portb= &b10000100
```

اعداد از سمت راست به چپ وضعیت ولتاژ پایه‌های B را مشخص می‌کنند و چون عدد سوم و هشتم را برابر یک قرار دادید پایه‌های PB2 و PB7 دارای ولتاژ می‌شوند.

حال اگر بخواهیم بوسیلهٔ دو کلیدی که به میکروکنترلر وصل کرده‌ایم اعداد 7-segment را کم و زیاد کنیم کد زیر را تایپ می‌کنیم.

```
$ regfile= "M8def.dat"
    Ddrd = &b11111111
    Ddrb = &b11000000
    Portb = &b11000000
    DIM S AS Integer
    S=0
    Declare sub Incremental
```

```
    Declare sub Decremental
Do
    IF pinb.6 = 0   Then   incremental
    IF pinb.7 = 0   Then   Decremental
    Select case S
    Case 0
    Portd = &b00111111
    Case 1
    Portd = &b00000110
    Case 2
    Portd = &b01011011
    Case 3
    Portd = &b01001111
    Case 4
    Portd = &b01100110
    Case 5
    Portd = &b01101101
    Case 6
    Portd = &b01111101
    Case 7
    Portd = &b00000111
    Case 8
    Portd = &b01111111
    Case 9
    Portd = &b01101111
END SELECT
Loop
Sub Incremental
    S = S+1
    If S>9 Then S=9
    Waitms 100
    Return
END Sub
Sub Decremental
    S = S – 1
    If S<0   Then S = 0
    Waitms 100
    Return
```

```
END Sub
```

در برنامه بالا از دستور جدیدی استفاده کردیم بنام Select case که بصورت کلی زیر نوشته می‌شود:

```
SELECT CASE
Case 1
.....
.....
case 2
.....
.....
case 3
.....
.....
case 4 to 9
.....
.....
case is>9
.....
.....
END SELECT
```

اگر در حین اجرای برنامه S مقدار ۱ را دارا باشد خطوط زیر Case 1 اجرا می‌شود اگر S مقدار ۲ را دارا باشد خطوط زیر Case 2 اگر مقدار S بین ۴ تا ۹ باشد مثلاً ۷ خطوط زیر Case 4 to 9 و اگر مقدار S بزرگتر از ۹ باشد خطوط زیر Case is>9 اجرا می‌شود شما بستگی به برنامه‌ای که می‌نویسید از هر کدام از این حالت‌ها می‌توانید بدلخواه استفاده کنید.

اما توسط دستور Declare Sub ما در زیر برنامه تعریف کردیم که به دلخواه نام آنها را incremental و Decremental قرار داریم هر بار در برنامه از نام این زیر برنامه‌ها استفاده می‌کنیم. نرم‌افزار به کد همان زیربرنامه رفته آن را اجرا می‌کند و توسط دستور Return که در انتهای زیربرنامه نوشته‌ایم نرم‌افزار دوباره به برنامه اصلی برمی‌گردد که زیربرنامه را معمولاً در آخر برنامه می‌نویسند.

حال به برنامه چشمک‌زن برمی‌گردیم و کد برنامه را طوری تنظیم می‌کنیم که با فشار دادن کلیدها سرعت چشمک‌زن LED و اعداد روی 7-segment با هم کم و زیاد شوند.

```
$ regfile = "M8def.dat"
    Ddrd = &b11111111
```

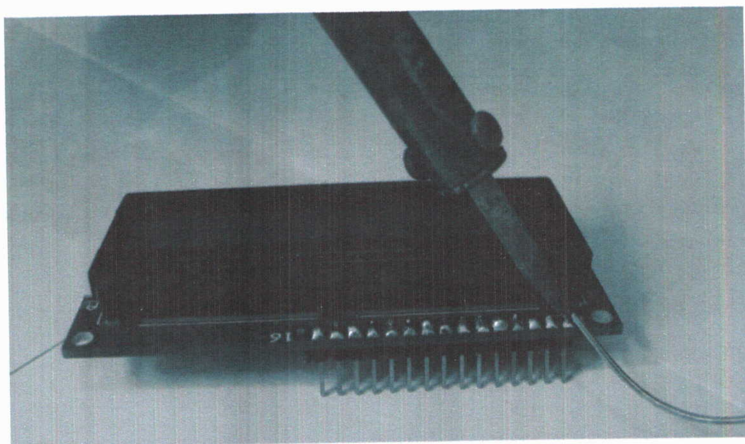
```
    Ddrb = &b11000000
    DIM S AS Interger
    S = 100
    Declare sub incremental
    Declare sub Decremental
Do
    Portb.2 = 1
    Waitms S
    Portb.2 = 0
    Waitms S
IF Pinb.6 = 0   Then   incremental
IF Pinb.7 = 0   Then   Decremental
    SELECT case S
    Case 0 to 9
    Portd = &b00111111
    Case 10 to 19
    Portd = &b00000110
    Case 20 to 29
    Portd = &b01011011
    Case 30 to 39
    Portd = &b01001111
    Case 40 to 49
    Portd = &b01100110
    Case 50 to 59
    Portd = &b01101101
    Case 60 to 69
    Portd = &b01111101
    Case 70 to 79
    Portd = &b00000111
    Case 80 to 89
    Portd = &b01111111
    Case 90 to 100
    Portd = &b01101111
END SELECT
Loop
    Sub Incremental
    S = S + 1
    IF   S> 100   Then   S = 100
    Waitms 100
    Return
END Sub
```

```
Sub Decremental
    S = S – 1
    IF  S<0   Then   S = 0
    Waitms 100
    Return
END Sub
```

توسط 7-segment توانستیم اطلاعات محدود عددی را نمایش دهیم حال اگر بخواهیم اطلاعات وسیعی از قبیل حروف، علامت‌ها و اعداد چندرقمی را نمایش دهیم باید از یک LCD استفاده کنیم.

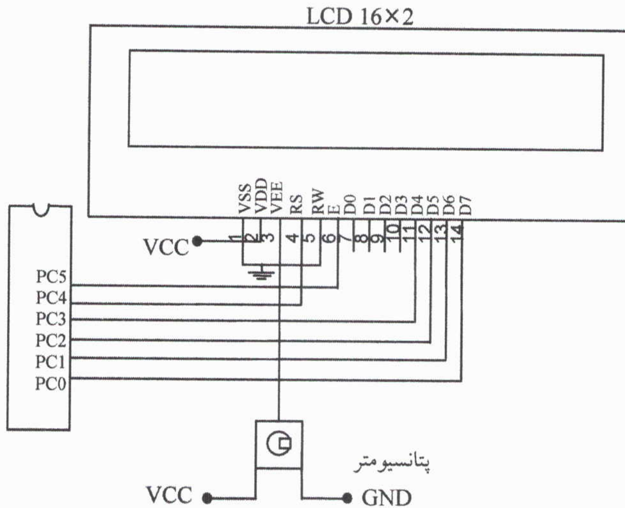
در این قسمت طریقه اتصال یک LCD (16×2) را به میکروکنترلر شرح می‌دهیم (منظور از ۲×۱۶ آن است که LCD ما شامل ۱۶ ستون و ۲ سطر می‌باشد).

برای آنکه بتوانیم LCD را بر روی بورد نصب کنیم ابتدا باید یک نرگی Pin Header را به آن لحیم کنیم برای این کار ابتدا ۱۶ پین از یک نرگی Pin Header را انتخاب کرده و با یک سیم چین آن را از بقیه جدا می‌کنیم و از طرف پین کوتاه داخل ۱۶ سوراخ LCD کرده و مطابق شکل لحیم می‌کنیم.



تصویر شماره ۳۲

و سپس نرگی Pin Header را داخل بورد به فاصلهٔ ۲ تا ۳ سانتی‌متر از میکروکنترلر فـرو می‌کنیم و سپس طبق نقشه زیر هر کدام از پایه‌های LCD را به میکروکنترلر وصل می‌کنیم:



پایه‌های شمارهٔ ۱ و ۵ به GND (-)

پایه شمارهٔ ۲ به منبع تغذیه (+)

پایه شمارهٔ ۳ به یک پتانسیومتر ۱۰ K

پایه شمارهٔ ۴ به PC4

پایه شمارهٔ ۶ به PC5

پایه شمارهٔ ۱۱ به PC3

پایه شماره ۱۲ به PC2

پایه شماره۱۳ به PC1

پایه شماره ۱۴ به PC0 وصل می‌شوند.

پتانسیومتر قطعه‌ای الکترونیکی است که درست مانند یک رئوستا عمل می‌کند یعنی با چرخاندن پیچ آن مقاومت آن کم و زیاد می‌شود و ما در اینجا برای تنظیم نور LCD از آن استفاده می‌کنیم.

کد زیر را برای شناساندن LCD به میکروکنترلر در ابتدای برنامه می‌نویسیم:

```
Config LCDPin = Pin , Db4 = Pin C.3 , Db 5 = Pin C.2 , Db6 = Pin C.1 , Db7 =
Pin C.0 , E = Pin C.5 , RS = Pin C. 4

Config LCD = 16 × 2
```

حال برای آنکه در برنامه LED چشمک‌زن بتوانیم مقدار متغیر S را توسط LCD نمایش دهیم که زیر را تایپ می‌کنیم:

```
$ regfile = "M8def.dat"
config LCD Pin = Pin, Db4 = Pin C.3 , Db5 = Pin C.2 , Db6 = Pin C. 1 , Db7 = Pin
C.0 , E = Pin C.5 , RS = Pin C.4

    config LCD = 16 × 2
    DDRb = &b11000100
    Portb = &b11000000
    DIM S AS Interger
    S = 100
Do
    Portb.2 = 1
    Waitms S
    Portb.2 = 0
    Waitms S
    CLS
    LCD "S variable="
    LCD S
 IF Pinb.6 = 0   Then    S = S + 1
 IF Pinb.7 = 0   Then    S = S – 1
 IF S<0 Then S = 0
 Loop
```

توسط خط LCD "S variable=" جملهٔ S variable بر روی LCD نقش می‌بندد و توسط دستور LCD S مقدار عددی S در جلوی جمله فوق نوشته می‌شود. مانند: S variable=100

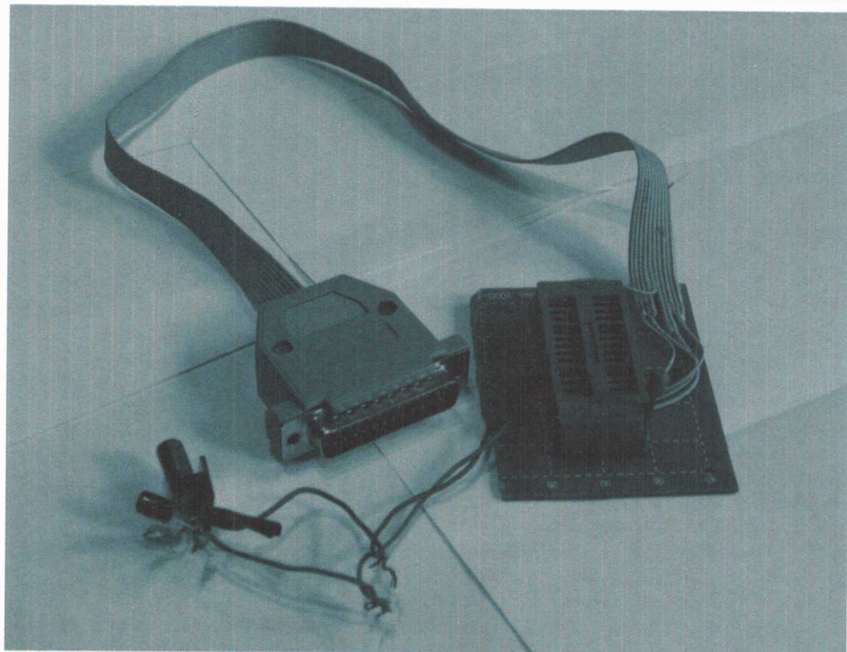
دستور CLS هم هر بار LCD را پاک می‌کند.

توسط خط IF S<0 Then S=0 به S اجازه نمی‌دهیم مقدار کمتر از S را بگیرد.

در حین کار کردن با قطعاتی که با تعداد سیم زیاد به میکروکنترلر وصل می‌شوند (مانند LCD و 7-segment) برای آنکه بورد خلوت شود ابتدا میکروکنترلر را خارج از بورد برنامه‌ریزی کرد سپس آن را بر روی بورد سوار نمود تا دیگر به ۵ سیمی که از انتهای کابل رابط به میکروکنترلر وصل شده احتیاجی نباشد برای اینکار به قطعات زیر احتیاج دارید:

- زیپ سوکیت ۲۸ پین
- یک بورد نقطه‌ای کوچک
- سیم فلت
- نرگی سوکیت ۲۵ پین به همراه کاور
- رگلاتور ۷۸۰۵
- خازن ۲۵ ولت، ۱۰۰ میکروفاراد
- خازن ۱۶ ولت ۴۷ میکروفاراد

نرگی سوکیت ۲۵ پین را مطابق دستور کابل رابط به سیم فلت وصل می‌کنیم زیپ سوکیت را بر روی برد نقطه‌ای لحیم کرده و انتهای دیگر سیم فلت را به پایه‌های zip سوکیت وصل می‌کنیم (توجه کنید که مثلاً سیم اول فلت که SKC است را به پایهٔ ۱۹ زیپ سوکیت وصل کنید چرا که زمانی که میکروکنترلر بر روی آن قرار می‌گیرد پایه ۱۹ زیپ سوکیت به پایه SKC و یا همان پایه ۱۹ میکروکنترلر وصل می‌شود همین‌طور بر پایه‌های دیگر توسط رگلاتور و خازن‌ها هم می‌توانید یک منبع تغذیه سرپایی درست کنید و پایه‌های خروجی آن را به پایه‌های ۷ و ۸ زیپ سوکیت که قرار است پایه‌های VCC و GND میکروکنترلر به آن وصل می‌شود، لحیم کنید حال می‌توانید منبع تغذیه را به یک آداپتور وصل کرده نرگی سوکیت ۲۵ پین را داخل پورت Printer کرده و یک میکروکنترلر به zip سوکیت سوار کنید مانند مراحل قبل می‌توانید میکروکنترلر را برنامه‌ریزی کرده سپس از روی زیپ سوکیت باز کرده و بر روی بورد سوار کنید.



تصویر شماره ۳۳

WatchDOG?

به برنامه زیر توجه کنید:

```
$ regfile = "M8def.dat"
  Dim A As integer
  A = 1
  Do
  A=A×2
  LCD A
Loop
```

در این برنامه میکروکنترلر پس از اجرا کردن خط اول، دوم و سوم به حلقه Do-Loop می‌رسد و تا زمانی که میکرو هنگ نکند دستورات بین Loop و Do را اجرا می‌کند یعنی هر بار مقدار A را در دو ضرب کرده در خودش می‌ریزد و آن را نمایش می‌دهد.

حال فرض کنید می‌خواهید برنامه را طوری تنظیم کنید که وقتی A مقدارش بزرگتر از عدد ۲۰۴۸ شد میکروکنترلر ری‌ست شود یعنی برنامه دوباره از خط اول شروع به کار کند. برای این کار از دستور WatchDOG استفاده می‌کنید. بدین صورت که توسط دستور:

```
Config watchDOG = 32
```

به میکروکنترلر می‌گویید که می‌خواهید ۳۲ میلی‌ثانیه بعد از دیدن دستور:

```
START WatchDOG
```

میکروکنترلر را ری‌ست کند.

حال برنامه به صورت زیر درمی‌آید:

```
$ regfile = "M8def.dat"
DIM A As interger
A = 1
Config WatchDOG = 32
Do
A= A×2
LCD A
IF A>2048  Then  START WatchDOG
Loop
```

توجه کنید وقتی مقدار A بزرگتر از ۲۰۴۸ شد برنامه وارد شرط IF شده و دستور START WatchDOG اجرا می‌شود و ۳۲ میلی‌ثانیه بعد میکروکنترلر ری‌ست شده و برنامه دوباره از خط اول شروع به اجرا می‌کند.

به دستوری مانند WatchDOG که می‌تواند روند اجرای برنامه را قطع کند و کار دلخواه دیگری را برایم انجام دهد وقفه یا interrupt می‌گویند. حال با interruptهای دیگری در میکروکنترلر آشنا می‌شویم.

Interrupt?

فرض کنید برنامه زیر را نوشته‌اید:

```
$ regfile = "M8def.dat"
Dim I As integer
i = 0
Do
i = i + 1
LCD i
Loop
```

وقتی چنین برنامه‌ای را اجرا می‌کنید i به ترتیب مقدارهای ۰، ۱، ۲، ۳ و... را گرفته و LCD آنها را نمایش می‌دهد و این روند تا جایی که حافظه میکرو اجازه می‌دهد ادامه پیدا می‌کند. اما گر بخواهید این روند را قطع کنید، فرض کنید می‌خواهید با زدن یک دکمه مقدار i دوباره صفر شود.

برای این کار احتیاج دارید برای میکرو تعریف کنید هر وقت به یک پایه خاص ولتاژ دادید روند اجرای برنامه را متوقف کرده، کار دلخواه ما را انجام دهد و دوباره به روند اجرای برنامه باز گردد.

حال به برنامه بالا دستورات جدیدی اضافه می‌کنیم:

ابتدا پایه‌ای از میکرو که نام int0 دارد را به میکرو معرفی می‌کنیم:

(فقط پایه‌هایی که نام int دارند به درد کار ما می‌خورند)

```
Config int0 = rising
```

و توسط کلمهٔ rising به میکرو می‌گوییم هر وقت ولتاژ ورودی به پایه int0 شروع به زیاد شدن کرد کار دلخواه ما را انجام بده. در حالتی که ولتاژی به پایه وصل نیست پایه ولتاژ صفر را دریافت می‌کند حال اگر یک ولتاژ ۵ ولتی به پایه وصل کنید چون ولتاژ از ۵ به صفر

تغییر می‌کند. میکرو کار دلخواه ما را انجام می‌دهد.*

اما کار دلخواه ما چگونه انجام می‌شود.

```
Config int0 = rising
```

توسط این دو دستور وضعیت پایه به حالت آماده‌باش درمی‌آید:

```
{ Enable interrupts
{ Enable int0
```

```
On int0 Label1
Do
  i = i +1
  LCD i
Loop
Label1:
  i = 0
Return
```

و توسط دستور on int0 Label 1 به میکرو می‌گویید هرگاه ولتاژ پایه int0 زیاد شد روند اجرای برنامه را قطع کند و به Label1 بپرد و خطوط زیر آن را اجرا کرده و با استفاده از دستور return به برنامه بازگردد.

نکته مهم آن است فرض کنید زمانی که میکرو مشغول اجرای خط i = i +1 می‌باشد شما به int0 ولتاژ داده‌اید. در این حالت برنامه نه بر روی خط on int0 Label1 قرار دارد و نه بر روی :Label1 با این حال میکروکنترلر به طور سخت‌افزاری می‌فهمد شما به int0 ولتاژ داده‌اید و برنامه را متوقف کرده و به برچسبی که شما تعریف کرده‌اید می‌پرد.

---

*. بجای کلمه rising اگر از کلمه falling استفاده کنید پایه به هنگام کم شدن ولتاژ کار دلخواه ما را انجام می‌دهد.

Timer?

فرض کنید ساعت کامپیوتری شما فقط ثانیه را نمایش می‌دهد. یعنی از صفر تا ۵۹ و بعد از اینکه عدد به ۵۹ رسید دوباره صفر می‌شود.

حال ساعتی را تصور کنید که بجای آنکه با رسیدن به عدد ۵۹ دوباره صفر شود با رسیدن به عدد ۶۵۰۰۰ دوباره صفر می‌شود.

چنین ساعتی Timer داخل یک میکروکنترلر می‌باشد با یک تفاوت و آن اینکه در ساعت کامپیوتری شما وقتی مثلاً عدد از ۵ به ۶ تغییر می‌کند زمان گذشته دقیقاً ۱ ثانیه می‌باشد ولی در ساعت درون میکروکنترلر وقتی عدد از ۵ به ۶ تغییر می‌کند زمانی که گذشته به مراتب بسیار کمتر از یک ثانیه می‌باشد.

و اما اینکه این زمان دقیقاً چقدر است بستگی به شما دارد که برای ساعتتان تعریف کنید با چه سرعتی کار کند. وقتی در شروع برنامه دستور زیر را می‌نویسید:

```
Config Timer = Timer0 , Prescale = 64
```

در قسمت Timer = Timer0 به میکروکنترلر می‌گویید می‌خواهید از ساعت Timer0 استفاده کنید. (میکروکنترلرها معمولاً دارای چندین ساعت می‌باشند. بنابراین باید به میکرو بگویید می‌خواهید از ساعت مثلاً Timer0 و یا Timer1 استفاده کنید.)

و اما در قسمت Prescale = 64 به میکرو می‌گویید سرعت عوض کردن اعداد چقدر باشد. شما می‌توانید در جلوی دستور Prescale از اعداد ۱، ۸، ۶۴، ۲۵۶ و ۱۰۲۴ استفاده کنید. اگر از عدد ۱ در جلوی Prescale استفاده کنید ساعت با بیشترین سرعت ممکن اعداد را عوض می‌کند و همین‌طور اگر از ۱۰۲۴ استفاده کنید ساعت با کمترین سرعت اعداد را عوض می‌کند.

حال که با نوع صداکردن یک ساعت در میکروکنترلر آشنا شدید چند دستور برای آنکه بتوانید با یک ساعت کار کنید را به شما معرفی می‌کنم.

ابتدا در شروع برنامه با دستور:

```
Config Timer = Timer0, Prescale = 64
```

ساعت را آماده کار می‌کنیم. سپس توسط دستور START Timero ساعت شروع به کار می‌کند. حال اگر در هر لحظه بخواهیم بدانیم ساعت چه زمانی را نشان می‌دهد از کلمه TCNT0 استفاده می‌کنیم. اگر یک LCD به میکرو وصل کنید توسط دستور زیر می‌توانید مقدار عددی ساعت را مشاهده کنید.

```
LCD TCNT0
```

اما کاربرد مهم دیگری که یک Timer دارد استفاده از آن به عنوان یک interrupt می‌باشد.

اگر یادتان باشد در ابتدای این بخش گفتیم ساعت میکروکنترلر برخلاف ساعت دستتان به جای آنکه با رسیدن به عدد ۵۹ صفر شود با رسیدن به عدد ۶۵۰۰۰ صفر می‌شود. اگر در جایی از برنامه از دستور زیر استفاده کنید:

```
On Ovf0 Label1
```

هربار که ساعت صفر می‌شود اجرای برنامه متوقف شده و دستوراتی که به طور مثال در Label1 نوشته‌اید اجرا می‌شود البته برای استفاده از Timer به عنوان یک interrupt باید از دو دستور زیر برای شروع کار Timer استفاده کنید.

```
Enable interrupts
Enable Timer0
```

به مثال زیر توجه کنید:

```
$ regfile = "M8def.dat"
Config Timer = Timer0, Prescale = 1024
Enable interrupts
Enable Timer0
On ovf0 Label1
START Timer0
Do
  LCD TCNT0
Loop
Label1:
CLS
Return
```

وقتی چنین برنامه‌ای را اجرا می‌کنید عدد داخل ساعت (TCNT0) مرتب بـر روی LCD نمایش داده می‌شود ولی هر بار که عدد درون ساعت به ۶۵۰۰۰ رسـید و دوبـاره صـفر شـد برنامه به Label1 پریده و صفحه LCD را پاک می‌کند و دوباره به داخل Do-Loop برگشته و شروع به نمایش اعداد بر روی LCD می‌کند.

Bitwait ?

اگر بخواهید اجرای برنامه در یک خط خاص تا زمان یک کار مشخص مثلا زده شدن یک کلید متوقف بماند, از دستور bitwait **استفاده می کنید.**

مثال:یک کلید شاسی را در نظر بگیرید که یک سر آن به پورت vcc وسردیگر آن به پورت c.5متصل است.به پورت c.4نیز یکLED وصل کنیدو برنامه زیرا به میکرو منتقل کنید.

```
$regfile="M8def.dat"
DDRC=&b11111111
Do
Bitwait pinc.5,set
Portc.4=1
Loop
```

پس از اجرای برنامه مشاهده می کنید که LED تا زمانی که شما کلید شاسی را فشار نداده اید,خاموش می ماند, یعنی اجرای برنامه در خط ۴ متوقف شده است,زمانی که شما کلید را فشار میدهید,مقدارPINC.5 یک شده و برنامه به خط بعدمی رودوlED روشن میشود.

اگر بجای کلمه SETاز کلمه RESETاستفاده کنید**زمانی که PINC.5صفر شود از BITWAITرد میشود**

Alias?

فرض کنید مشغول انجام برنامه ای هستید که در آن میکرو به چند کلید و LEDوصل شده است و بر اساس زده شدن کلید هاlED ها خاموش و روشن می شود.در یک برنامه ساده کلید شماره ۱ را به PORTC.5 و LEDمثلاسبز رنگ را به PORTC.4وصل کنیدو برنامه را این گونه می نویسید.

```
$REGFILE="M8def.dat"
DDRC=&b11111111
Do
 If pinc.5=1 then
   portc.4=1
   waitms 100
   portc.4=0
 End if
loop
```

اما این برنامه رابه گونه دیگری هم می توان نوشت:

```
$REGFILE="M8def.dat"
DDRC=&B11111111
  Key1 Alias pinc.5
  Sabz  Alias Portc.4
Do
  If key1=1 then
     Sabz=1
     Waitms 100
     Sabz=0
  End if
Loop
```

در حقیقت شما برای برنامه تعریف کرده اید که از این به بعد بجای pinc.5 از کلمه key1

بجای portc.4 از کلمه sabz استفاده کند.این روش به درد برنامه هایی می خورد که شما از port و pin زیاد استفاده می کنیدو می خواهید درحین نوشتن برنامه,انها را با هم قاطی نکنید.

EEPROM ?

EEPROM نام حافظ ای از میکرو کنترلر است که دقیقا مانند HardDisk یک کامپیوتر عمل می کند یعنی با خاموش روشن کردن میکرو حافظه پاک نمی شود.

> این بدان معنی نیست که با Program کردن دوباره میکرو حافظه پاک نشود.

مثال:

```
$regfile="m8def.dat"
DDRD=&b11111111
Dim b as byte
Dim x as byte
B=3
Do
    Writeeprom b,2
     Readeepromx,2
     portd.x=1
loop
```

در خط۷ عدد ۳ در حافظهء۲ ذخیره می شودو در خط ۸ حافظهء۲ خوانده می شودودرون متغیرx ریخته می شود و LED که به Portd.3 متصل است روشن می شود.

رله چیست؟

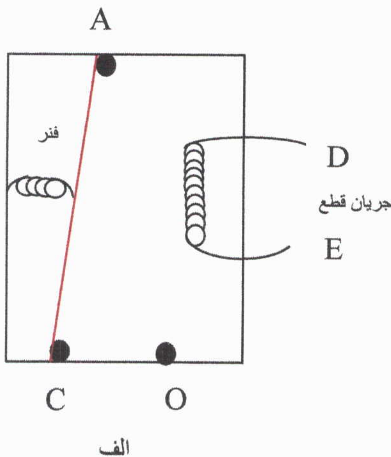
رله بوبینی(سیم پیچی)است که وقتی به آن جریان می دهیم تبدیل به آهنربا شده ویک تکه سیم را به طرف خود می کشد. در شکل الف جریان دو سر بوبین DE قطع است ، بنابراین فنر تکه سیم قرمز را به سمت خود میکشد.در نتیجه نقطهA به نقطهC وصل است .

NC:normal close اما زمانی که جریان دو سر بوبین وصل است بوبین تبدیل به آهنربا شده و تکه سیم قرمزرا به سمت خود می کشد،در نتیجه نقطه A به نقطهO وصل می شود.

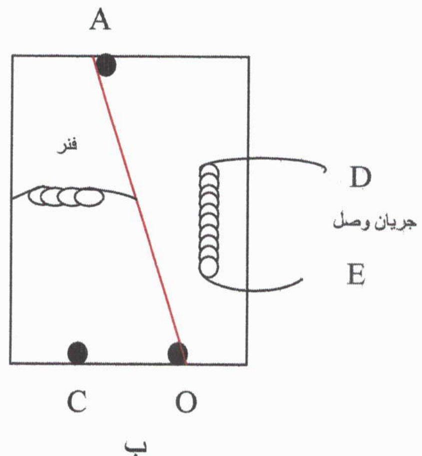
NO:normal open

منظور از NO NC این است که در حالت عادی که جریان در بوبین نیست نقطهc بسته و نقطهO باز است.

وقتی می گوییم یک بوبین رله ۱۲ ولت یعنی برای آنکه رله عمل کند باید ۱۲ ولت به بوبین ولتاژ بدهیم تا بوبین اهنربا شود و سیم قرمز را به طرف خود بکشد.یک رله ۵ ولت به ۵ ولت نیاز دارد تا به آهنربا تبدیل شود.

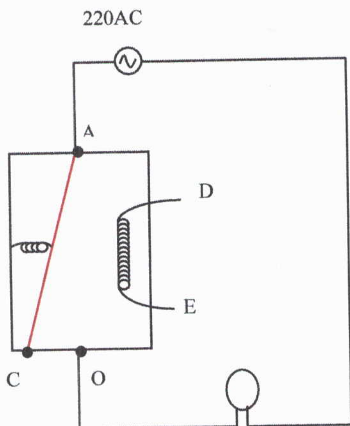


الف



ب

اما رله در چه مواردی استفاده می شود؟
فرض کنید می خواهید توسط یک میکروکنترلر یک لامپ ۲۲۰ ولتی را روشن و خاموش کنید.در این حالت باید از یک رله استفاده کنید.

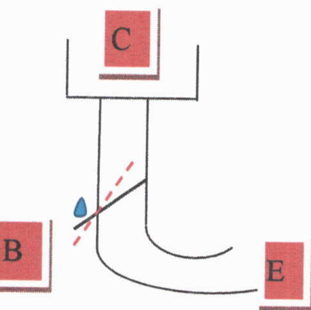


با توجه به شکل هر گاه جریان دو سر بوبین و قطع باشد مدار لامپ ۲۲۰ ولتی باز بوده ولامپ خاموش است, اما وقتی به بوبین جریان ۵ ولت Dc وصل می کنیم , بوبین آهنربا شده وAبهO وصل می شود.در این حالت مدار لامپ بسته شده و لامپ روشن می شود .
اما برای وصل کردن بوبین DE به یک میکرو کنترلر, ساده ترین راهی که به ذهن ما ن می رسد آن است که نقطه D را به یکی از Port های میکروو نقطه E را به GND (زمین)وصل کنیم وبا یک کردن Port مورد نظرروشن و خاموش لامپ را کنترل کنیم, اما این کار یک اشکال دارد, بوبین ها معمولا بیشتر از توان یک میکرو جریان می کشند , در این حالت میکرو مرتبا reset شده وقادربه اجرای برنامهءمورد نظر نمی باشد.
اما راه حل, استفاده از یک ترانزیستور می باشد.

ترانزیستور

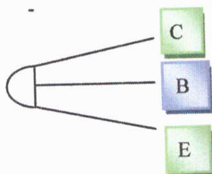
اغلب وقتی برای یاد گرفتن ترانزیستور و کار کردن با آن به سراغ کتابهای الکترونیک می رویم، پس از صرف زمان نسبتا طولانی تنها اطلاعاتی راجع به ارایش الکترونی بعضی عناصر، نیمه هادی ها و جریان الکتریکی بین آنها بدست می آوریم. اما خب بعد از آن باز هم از استفاده از یک ترانزیستور عاجز می باشیم.!

یک منبع آب را تصور کنید که توسط یک لوله قطور آب از آن خارج می شود،در وسط این لوله اهرمی قرار دارد که جریان آب را قطع می کند.این اهرم به گونه ای طراحی شده است که اگر یک قطره آب روی آن بچکد،
برای لحظه ای دریچه را باز می کند و
بعد از آن دوباره دریچه بسته می شود.
حال اگر متناوبا قطرات آب روی آن
بچکددریچه مرتبا باز و بسته می شود.
ما اسم منبع آب را Collector، اسم
خروجی لوله را Emitor واسم اهرم
دریچه راBase میگذاریم.

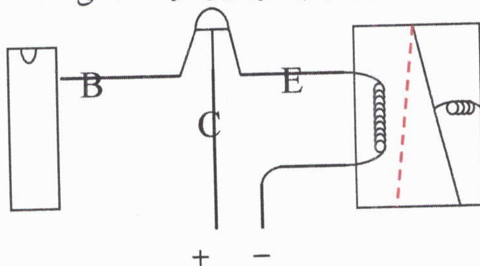


توضیح منبع آب بدین صورت در می آید:
هر گاه جریان کوچکی از آب را به base بدهیم دریچه باز شده و جریان بسیار بزرگی (متناسب با ارتفاع منبع آب وقطر لوله اصلی)از collector به emitor می رود.

حال یک ترانزیستور را با ۳ پایهء آن تصور کنید.نام ۳پایه ترانزیستور هم base,collector emitor, می باشد . حال اگر collector را به یک منبع جریان نسبتا زیاد وصل کنیم , هر گاه به پایه base جریان بسیار کوچکی بدهیم, در آن صورت پایه collector به emitor متصل می شود و آن شدت جریان بالا به emitor می رود.



حال می خواهیم با استفاده از یک ترانزیستور 2n2222 یک رله را توسط میکرو کنترلر کنترل کنیم.آرایش زیر طرز بستن میکرو کنترلر ترانزیستور و رله را نشان می دهد.



در واقع base با جریان کوچکی که از میکرو می گیردcollectorرا که به یک منبع جریان بیشتر متصل است را به emitor وصل می کند.

اما برای تشخیص پایه های ترانزیستور:

Multimeter خود را بر روی علامت ⟶ تنظیم کنیم در این حالت Multimeter عدد 0 را نشان می دهد.حال دو سیم مثبت و منفی Multimeter را دو به دو بر روی پایه های ترانزیستور قرار می دهیم اگر سیم منفی بر روی پایه base قرار گیرد ,مولتی متر هیچ عددی را نشان نمی دهد ولی اگر سیم مثبت بر روی پایه base قرار گیرد با هر بار گذاشتن سیم منفی بر روی دو پایه کناریMultemeterدو عدد سه رقمی را نمایش میدهد که عدد بزرگتر مربوط به پایه collector و عدد کوچکتر مربوط به پایه emitorمی باشد.

در این ضمیمه لیستی از قطعات مصرفی لوازم مورد نیاز تهیه شده است.

۲- میکروکنترلر Mega8

۳- 7-segment کاتد مشترک

۴- 2 × LCD 16

۵- LED

۶- کلید شاسی

۷- برد تخته‌ای (Bread board)

۸- مقاومت ۳۳۰ اهم

۹- رگلاتور ۷۸۰۵

۱۰- خازن ۲۵ V و ۱۰۰ µf (خازن ۲۵ ولت، ۱۰۰ میکروفاراد)

۱۱- خازن ۱۶ V و ۴۷ µf (خازن ۱۶ ولت، ۴۷ میکروفاراد)

۱۲- کانکتور مخابراتی ۵ پین

۱۳- نرگی سوکیت ۲۵ پین (به همراه کاور)

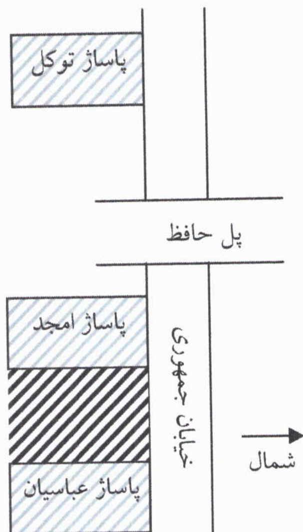
۱۴- پتانسیومتر (پاناسونیکی) ۱۰ K

۲۰- Pin Header نرگی

۲۱- سیم فلت ۱۰ رشته

۲۲- سیم برد بورد یا سیم مفتولی تلفن

۲۳- آداپتور با خروجی DC ولتاژ بیش از ۶ ولت.



شما قادر خواهید بود در قسمتهای هاشور خورده مواد مصرفی خود را تهیه کنید.

لوازم مورد نیاز:

۱- سیم چین کوچک

۲- هویه

۳- سیم لحیم

۴- multimetr

توجه کنید می‌خواهید دو لامپ LED را چشمک‌زن کنید. برای این کار دو راه دارید. ابتدا می‌توانید بوسیلهٔ دو خازن، دو ترانزیستور و چهار مقاومت مداری طراحی کنید که زمانی که دو LED را به آن وصل می‌کنید، آنها چشمک بزنند، مانند کیتهای آماده‌ای که در بازار برای این کار وجود دارد.

اما راه حل دوم آن است که به جای استفاده از قطعات فوق از دستورات برنامه‌نویسی درون یک میکروکنترلر استفاده کنید، دستورات برنامه‌نویسی توسط میکروکنترلر به طور تقریبی مدار سخت‌افزاری بالا را برای شما شبیه‌سازی می‌کنند و می‌توانید توسط یک حلقه و چند دستور سادهٔ DDR، Port، LEDها را چشمک‌زن کنید. اگر تنها چند خطی برنامه‌نویسی Basic می‌دانید و اطلاعات الکترونیکی ندارید کتاب فوق مراحل کار با میکروکنترلر را به شما آموزش می‌دهد.